Evolving S-Boxes with Reduced Differential Power Analysis Susceptibility

Merrielle Spain[1] and Mayank Varia[2†]

[1] MIT Lincoln Laboratory,[‡] merrielle.spain@ll.mit.edu
[2] Boston University, varia@bu.edu

Abstract.
Differential power analysis targets S-boxes to break ciphers that resist cryptanalysis. We relax cryptanalytic constraints to lower S-box leakage, as quantified by the transparency order. We apply genetic algorithms to generate 8-bit S-boxes, optimizing transparency order and nonlinearity as in existing work (Picek et al. 2015). We apply multiobjective evolutionary algorithms to generate a Pareto front. We find a tight relationship where nonlinearity drops substantially before transparency order does, suggesting the difficulty of finding S-boxes with high nonlinearity and low transparency order, if they exist. Additionally, we show that the cycle crossover yields more efficient single objective genetic algorithms for generating S-boxes than the existing literature. We demonstrate this in the first side-by-side comparison of the genetic algorithms of Millan et al. 1999, Wang et al. 2012, and Picek et al. 2015. Finally, we propose and compare several methods for avoiding fixed points in S-boxes; repairing a fixed point after evolution in a way that preserves fitness was superior to including a fixed point penalty in the objective function or randomly repairing fixed points during or after evolution.

Keywords: S-box · nonlinearity · transparency order · genetic algorithm · multiobjective evolutionary algorithm · cycle crossover · differential power analysis · cryptanalysis · trade-off · Pareto front

1 Introduction

Block ciphers are a versatile, foundational primitive within symmetric-key cryptography. They underpin symmetric-key encryption schemes [Dwo01], message authentication codes [Dwob], authenticated encryption schemes [Dwoa, Dwo07, Dwo10], and hash functions [PGV93, BRSS10].

Internally, a block cipher often contains a sequence of mostly-identical *rounds* that combine linear and nonlinear components. Many block ciphers provide nonlinearity by a table lookup into a *substitution box*, or S-box. Block ciphers have built on S-boxes for 40 years, dating back to the Digital Encryption Standard and other early ciphers [Nat77]. Since then, S-boxes have appeared in most Advanced Encryption Standard (AES) competition entries [DR02,BAK98,Sch93,SKW+99,BCD+98,Mas93,Lim99,KMTM00], ciphers based on AES that maintain its S-box [JNP14,HKR15,Bor00,Nik14], lightweight ciphers using smaller S-boxes [LPPS07,BCG+12,SMMK12,BKL+07], and more [DEMS14,BR00,AIK+00,Mer90].

[†] Research performed while consulting at MIT Lincoln Laboratory.

[‡] Distrubion A: approved for public release; unlimited distribution. This material is based upon work supported by the Department of the Navy under Air Force Contract No. FA8721-05-C-0002 and/or FA8702-15-D-0001. Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Department of the Navy.

Proper S-box design requires careful analysis of conflicting features. First, the designer must choose the *size* of the S-box, balancing cryptographic strength with performance. Second, the S-box must resist mathematical *cryptanalysis*; at a minimum, it should have high algebraic degree when expressed as a polynomial, and correlate poorly with any linear function. Third, the S-box should be amenable to a hardware implementation that can resist power-based *side channel attacks* [KJJ99]. Fourth, the S-box should lack any *fixed points* that might unravel the cipher's round structure.

While our approach is size-agnostic, this paper focuses on 8-bit permutations such as the S-box in AES and several other ciphers [Nat01, DR02, Mas93, KMTM00]; these S-boxes support practical speeds in hardware and software, yet their space of possibilities defies exhaustive search. By contrast, the space of 4-bit permutations is enumerable—only 302 S-boxes exist up to affine equivalence [DC07, pp. 88-94].

1.1 Related work

Researchers have been automatically generating S-boxes for a decade by using genetic algorithms [MBC+99, WWLL12, INN14, CF04]. Most of these approaches focused solely on the cryptanalytic strength of the generated S-box while ignoring side channel attacks.

Side channel metrics. Two factors govern the strength of power-based side channel attacks: the signal inherent in the cryptographic algorithm and the noise introduced by the implementation. While one can measure the latter empirically on a test harness, designing an objective function for a genetic algorithm requires analytical metrics. *Transparency order* [CSM+15] captures how well differential power analysis (DPA) [KJJ99] will compute the key given an infinite signal-to-noise ratio (SNR). In other words, it measures how much an ideal distinguisher would leak through Hamming weight [CSM+15]. Transparency order is only well-defined for balanced functions, which suffices for permutations. Recently, the confusion coefficient attempted to analytically capture the noise introduced by an implementation [FLD12].

Transparency order indicates that highly nonlinear functions leak more than linear functions. Correspondingly, prior works have demonstrated the difficulty of extracting secret material from linear operations like xor via a power analysis attack [Jaf07, CMS+14]. Indeed, S-boxes provide the popular weak point to attack with power analysis.

Automatic generation of S-boxes. Picek et al. relaxed the cryptanalytic constraints to design 8-bit S-boxes with improved side channel properties as measured by the original transparency order and the confusion coefficient [PEB+14, PPE+14]. Note that in the 4-bit case they could simultaneously maximize both cryptanalytic and side channel properties [PEP+14]. When its authors redefined the transparency order to compensate for oversights in the original definition [CSM+15], Picek et al. applied their genetic algorithms approach to the corrected transparency order [PMMB15]. They asserted that "heuristics like genetic algorithms are not appropriate" for larger (e.g., 8-bit) S-boxes. They found that their approach on larger S-boxes generated worse values for nonlinearity and that the modified transparency order took prohibitively long to evaluate.

Prior work either optimized a single criterion [MBC+99, WWLL12] or a weighted sum of criteria [CF04, PEB+14, PPE+14, PEP+14, PMMB15]; which makes sense for criteria of similar types, or when carefully weighting an understood trade-off. Work that combined cryptanalytic and side channel properties failed to explore the trade-off [PEB+14, PPE+14, PEP+14, PMMB15], other than to provide loose bounds [MMS13]. Instead, it focused on generating a few S-boxes with superior properties.

Ivanov et al.'s work on "reversed" genetic algorithms [INN14] applies multiple criteria as thresholds, but evades direct comparison. In genetic algorithms, selection gradually drives the population toward better individuals. Ivanov et al. define "acceptable" and generate children until they obtain enough acceptable individuals. We consider this search using several genetic operators instead of a genetic algorithm per se.

Theory of genetic algorithms. Foundational understanding of genetic algorithms arises from schema theory: short, low-order, above-average schemata become exponentially more prevalent [Hol68]. A schema is a template describing similarities at certain positions and wildcards elsewhere. As good building blocks become more prevalent they lead to good individuals, provided that the problem can be decomposed into relevant building blocks.

This building block concept applies when crossover is common and mutation is minimal, as is traditionally recommended [DJ75]. For frequent, disruptive mutations (e.g., [PEB+14, PPE+14, PMMB15]) schema theory may no longer apply, because even short schema can be frequently broken. In contrast, even when frequent, the swap mutation (Section 3) preserves structure.

Fixed points. A fixed point through a round function can negate the security provided by a multi-round cipher. Given flexibility, it seems prudent to avoid fixed points in S-boxes, as Rijndael's designers discuss [DR02]. Most previous work disregards fixed points [MBC+99, CF04, WWLL12, MMS13, PEB+14, PPE+14, PMMB15]. The exceptions use affine transformations to remove them from 4-bit S-boxes [PEP+14] or check for them in the final output [INN14]. Previous genetic algorithms work fails to include a fixed point penalty in the objective function or repair fixed points.

1.2 Our contributions

This paper makes four contributions. First, we describe the trade-off (empirically) between nonlinearity and transparency order in 8-bit S-boxes. To accomplish this, we apply multiobjective evolutionary algorithms to populate a Pareto front of S-boxes with high nonlinearity and low transparency order. Multiobjective optimization is a mature approach in genetic algorithms [DAPM00, ZLT01], but we are the first to apply it to S-box generation.

Second, we find a more efficient genetic algorithm for generating S-boxes by drawing from schema theory and choosing problem-relevant building blocks and genetic operators that preserve and propagate them. To accomplish this, we are the first to apply the cycle crossover [OSH87] to S-box generation.

Third, we provide the first side-by-side comparison of the genetic algorithms of Millan et al. [MBC+99] (framework reused [CF04]), Wang et al. [WWLL12], and Picek et al. [PMMB15] (same framework as in their other works [PEB+14, PPE+14]). We evaluate 1,000 independent 12,000-evaluation runs of each algorithm. This reveals the efficiency and expected performance of each algorithm. Existing work compares results with random search, instead of with other existing work.

Fourth, we propose and compare several methods for finding S-boxes without fixed points, such as including a fixed point penalty in the objective function or repairing fixed points during and after evolution. Previous genetic algorithms work neglects to constrain search to S-boxes that lack fixed points.

1.3 Organization

We structure the paper as follows: Section 2 investigates the empirical trade-off between cryptanalytic and side channel properties. Section 3 introduces a new algorithm and compares the efficiency of existing algorithms in generating S-boxes with good cryptanalytic and side channel properties. Section 4 explores several ways to avoid fixed points and investigates the impact of these approaches on other properties. Section 5 presents our conclusions.

2 Trade-off Between Cryptanalytic and Side Channel Properties

2.1 Pareto front

We generate a Pareto front to display the empirical trade-off between transparency order and nonlinearity. A Pareto front is the set of non-dominated individuals: for each individual, no previously seen individual beats them under some criteria and equals them under the remaining criteria. We populate this front with two multiobjective genetic algorithms [DAPM00, ZLT01] as implemented in the Distributed Evolutionary Algorithms in Python toolbox [FDG$^+$12]. SPEA2 and NSGA-II both primarily prefer non-dominated individuals and secondarily prefer low density regions. SPEA2 considers the set of individuals that dominate a solution and sums how many individuals each member of the set dominates; 0 for non-dominated points, and high for points dominated by many points or dominated by points that dominate many points. They estimate density with the distance to the kth nearest point. They combine these preferences such that being dominated always eclipses density. NSGA-II splits points by the front they belong to, and then orders points belonging to the same front by density. They measure density with distance to opposing neighbors along each objective. Past work has found that the two approaches behave similarly [ZLT01].

Rijndael variants score optimal cryptanalytic values, and affine functions score optimal transparency order. Hence we seed these experiments with these individuals as well as random ones; like Ivanov et al. [INN14], we start these optimizations in informed locations. We used publicly available code for fast generation of Rijndael variant S-boxes [SV16]. We apply the cycle crossover and swap mutation described in Section 3. The children result from either crossover or mutation. The probability that a child is generated by crossover is 0.7 and generated by mutation is 0.3. We apply a $(\mu + \lambda)$ evolutionary algorithm: it selects the next generation from both the children and the parents. The population size is 20, and we run 1,000 independent trials each for 2,000 generations.

Specifically, we apply the Wilcoxon rank-sum test [Wil45] to determine whether SPEA2 and NSGA-II perform equivalently. We pair transparency order values based on nonlinearity values (ignoring unpaired values). We find that SPEA2 performs slightly, but significantly better ($p \ll 0.001$).

Figure 1 compares the multiobjective S-boxes with single objective S-boxes, manually designed S-boxes, and random permutations. The Cycle data are generated as described in Section 3 using the same crossover and mutation, but a single objective. The Picek data, included from previous work [PMMB15], also depend on a single objective. Cycle, the single objective equivalent of our multiobjective approach, covers less of the Pareto front, but achieves better individuals on nonlinearity values where it produces results.

We also include the manually-designed 8-bit S-boxes from the AES [DR02], SAFER [Mas93], Camellia [AIK$^+$00], and E2 [KMTM00] ciphers. The manually-designed S-boxes display values similar to, but worse than the multiobjective S-boxes.

To identify typical values for random permutations, we generate 10,000 random permutations and measure their transparency order and nonlinearity. The random permutations concentrate tightly in the shaded area. As described in Section 3, the Picek and Cycle algorithms start from a population of random permutations. While the results may be far from the likely region relatively, they are near absolutely. That is, while in low probability regions, they score similarly to random permutations.

2.2 Data requirements for linear cryptanalysis

To understand the cryptanalysis part of the trade-off, this section calculates the number of plaintext/ciphertext pairs needed to break a cipher using linear cryptanalysis based on

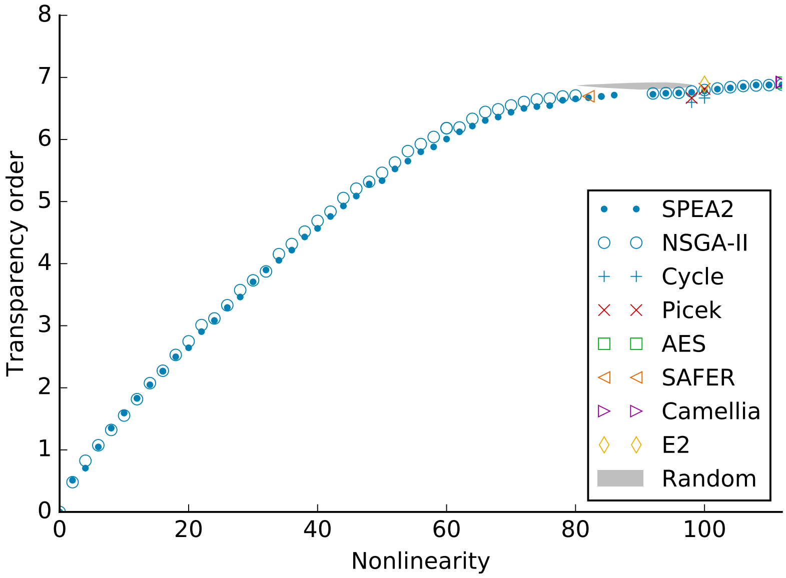

Figure 1: The Pareto front of the best individuals generated by the SPEA2 (blue dot) and NSGA-II (blue circle) multiobjective genetic algorithms displays the trade-off between transparency order and nonlinearity. Cycle (blue plus) and Picek et al. [PMMB15] (red cross) provide single objective comparisons. Manually-designed 8-bit S-boxes from the AES (green square), SAFER (orange triangle), Camellia (purple triangle), and E2 ciphers (yellow diamond) provide additional comparisons. The most probable scores for random permutations (gray region) also appear.

Table 1: Lower bound on the number of plaintext/ciphertext pairs D required to attack an S-box with nonlinearity NL via linear cryptanalysis with 98% probability, assuming that the cipher follows the round structure of AES.

NL	112	86	75	64	54	44	24
D	2^{300}	2^{160}	2^{128}	2^{100}	2^{80}	2^{60}	2^{30}

S-box nonlinearity. Notice that nonlinearity is a property of an S-box whereas resistance to linear cryptanalysis is a property of the entire cipher. While there exist methods to estimate the data requirements for general ciphers [BG09, BGT11], this section assumes that the S-box is embedded in a cipher whose round structure matches AES so that we may inherit the following theorem.

Theorem 1 (Wide-trail strategy). *Let $c_S = 1 - \frac{NL}{128}$ denote the correlation between the inputs and outputs of a given S-box, and let C_4 denote the maximum correlation contribution of a linear trail through 4 rounds of the cipher. Then, $C_4 \leq c_S^{25}$.*

Applying this theorem twice immediately yields a maximal correlation of $C \leq c_S^{50}$ through 8 rounds of the cipher. We assume generously that the remaining rounds of the cipher cause no additional complications.

Next, let D denote the number of plaintext/ciphertext pairs at the attacker's disposal. Observe that linear cryptanalysis attempts to solve a system of linear equations that is overconstrained (which is good for the attacker) but such that each equation is only accurate with probability C (which is bad for the attacker). Linear algebra shows that the attacker requires $D = O(C^{-2})$ plaintext/ciphertext pairs in order to solve this system with constant probability [KR11, §8.2]. In particular, $D \geq C^{-2} \geq \left(\frac{128}{128-\text{NL}}\right)^{100}$ samples suffice to solve the system with probability greater than 98%.

Concretely, Table 1 shows the nonlinearity required by an S-box in order to achieve certain lower bounds on the data complexity of an attack. We observe that the high data complexity bounds provide value to S-boxes whose nonlinearity is even *worse* than random, justifying our multiobjective approach to study the entire Pareto front.

3 Algorithm Efficiency and S-box Quality

Genetic algorithms consist of three operations: selection according to fitness, crossover, and mutation. Table 2 describes previous approaches' choices for these operations. They fall into the three general approaches of Millan et al. [MBC+99], Wang et al. [WWLL12], and Picek et al. [PMMB15]. We substitute their fitness functions with that of Picek et al. [PMMB15] for these experiments. Short descriptions of the operations follow; the corresponding works present longer descriptions.

Selection Selection eliminates weaker individuals, strengthening future generations. The relevant literature includes two selection methods. The first chooses the best k individuals out of the population to survive. Millan et al. and Wang et al. select individuals in this way. The second randomly selects three individuals without replacement; the better two survive and reproduce such that their child replaces the worst of the three. Picek et al. and Cycle select the best two of three for survival.

Crossover Crossover combines traits from multiple parents into children in structured, randomized recombination. If two parents excel in independent ways, then a child might inherent both and be exceptional. Partially matched crossover and order crossover are

Table 2: Previous genetic algorithm approaches for generating S-boxes. The literature provides three existing frameworks. This work introduces the approach named Cycle. Note that Millan et al. perform swap in a hybrid approach; their genetic algorithm lacks a mutation step.

Approach	Crossover	Mutation	Fitness
Millan et al. [MBC+99]	Assimilation	(Swap)	Nonlinearity or autocorrelation
[CF04]	Assimilation	(Swap)	Nonlinearity and difference uniformity
Wang et al. [WWLL12]	Exchanges bits	Swap length 10	Nonlinearity
Picek et al. [PMMB15]	Partially matched Order	Insert Inversion	Nonlinearity and 2015 transparency order
[PEB+14]	Partially matched Position-based Order	Insert Inversion Swap	Nonlinearity and 2005 transparency order
[PPE+14]	Partially matched Position-based Order	Insert Inversion	Nonlinearity and confusion coefficient
Cycle (this work)	Cycle	Swap	Nonlinearity and 2015 transparency order

classic crossovers for permutation individuals; the first respects absolute order, while the second respects relative order. We used preexisting implementations of both [FDG+12]. Assimilation [MBC+99] fills values into a child by choosing randomly between the parents for each location. When the value from neither parent remains available (to maintain bijectivity), a random unused value is chosen. Wang et al. exchange all the ith bits in one parent for all the jth bits in the other parent.

The cycle crossover iteratively finds pairs of parental values by position to preserve bijectivity [OSH87]. These cycles consider both parents, so they differ from cycles found within a single permutation. This approach preserves a large amount of structure from each parent. In fact, all structure comes from one of the parents. In contrast, assimilation randomly chooses many later values when neither parental value remains available; this introduces unrelated structure. Partially matched crossover exchanges chunks of values. To compensate for differences in the value sets, reverse substitutions occur elsewhere. This introduces structure unrelated to both parents. Order crossover preserves relative order in a permutation by sliding entries; while sensible for the traveling salesman problem, it makes less sense for S-boxes where the input-output mappings matter. Exchanging bits requires extensive repair to achieve a permutation; this introduces unrelated structure.

Mutation Mutation is traditionally considered an insurance policy against the loss of genetic material [Gol89]. A swap mutation replaces the values at two positions with each other. It is possible to interchange sequences [WWLL12]. An insert mutation moves a single value from one location to another, shifting other values to compensate. An inversion mutation reverses the order of a sequence.

Table 3: Genetic algorithm parameters. We ran 1,000 independent trials for each algorithm. These parameter choices approximate those of the original implementations, while reaching 12,000 evaluations.

Approach	Parents	Children	Generations	Mutation rate
Millan et al. [MBC+99]	10	45	267	-
Wang et al. [WWLL12]	10	40	300	0.5
Picek et al. [PMMB15]	50	16	750	0.3
Cycle (this work)	20	6	2,000	0.3

3.1 Comparison of Genetic Algorithms

We implemented the three existing frameworks along with our own, and we tested them as uniformly as possible. Table 2 lists the selection, crossover, and mutation type for each algorithm. For homogeneity, we replaced the fitness functions with a single objective summing nonlinearity and 2015 transparency order, as applied in Picek et al. [PMMB15]. Millan et al.'s non-hybrid approach was used. For Wang et al., we replace the generation of parameters from chaotic systems with pseudorandom numbers, as in the other algorithms.

Millan et al. prevented repeat individuals in the population pool. This traditional choice prevents local optima from trapping the algorithm [Gol89]. We found that this improved performance and applied it to all the algorithms. Table 3 describes the parameters for each algorithm. Wang et al. generate 4,000 children each in more than 1,500 generations, which we found infeasible for the measurement cost of the 2015 transparency order. Instead we pull their number of evaluations into line with the others. While Picek et al. enable early termination, we maintain a consistent number of generations. We run each algorithm to reach 12,000 fitness function evaluations.

Figure 2 shows how the transparency order of the best individual in a run evolves over time. The best individual in a population has the best fitness value (typically the best transparency order of the individuals with the best nonlinearity). We measure time in fitness function evaluations; generations would be unfair as the approaches differ in parent populations and child ratios. For the 1,000 independent runs of each algorithm, we show the median and range. The range indicates the best and worst values of the 1,000 best individuals. The Cycle approach outperforms existing approaches at quickly reducing the transparency order. The best runs of Millan near the median runs for Cycle. The best runs for Picek and Wang reach the worst runs for Cycle. The small increases in transparency order for Picek and Millan correspond to increases in nonlinearity.

Figure 3 shows how efficiently the algorithms improve nonlinearity. The coarse-grained structure of nonlinearity leads us to evaluate performance differently. The algorithms achieved few nonlinearity values, the best of which was 100. We consider all runs, of 1,000, that reach a nonlinearity of 100.[1] We rank the runs within an algorithm, such that we compare the most efficient runs with each other, the second most efficient runs with each other, and so forth. Wang failed to achieve a nonlinearity of 100. Cycle and Millan consistently outperform Picek. While less efficient than Cycle, more Millan runs achieve a nonlinearity of 100. Note that our later comparison between Cycle and Millan changes when we consider more individuals per run and exclude S-boxes with fixed points.

Figures 2 and 3 indicate how efficiently an algorithm achieves a good solution. The efficiency figures describe the strength of the best individual over time. Some applications may require a large pool of good individuals. Now, we compare the distributions of good individuals generated by each algorithm.

[1] Technically we measure when the best individual reaches a nonlinearity of 100, but these are equivalent given the fitness function and our knowledge of the pareto front.

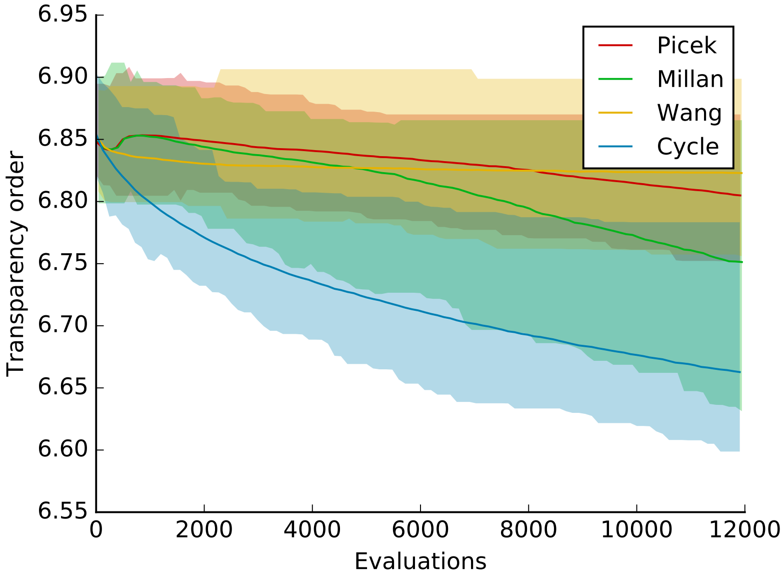

Figure 2: Transparency order of best individual as a function of number of evaluations (lower is better). The median of 1,000 independent runs (line) appears, as well as the minimum and maximum (shaded region). The Cycle approach outperforms existing approaches at reducing the transparency order.

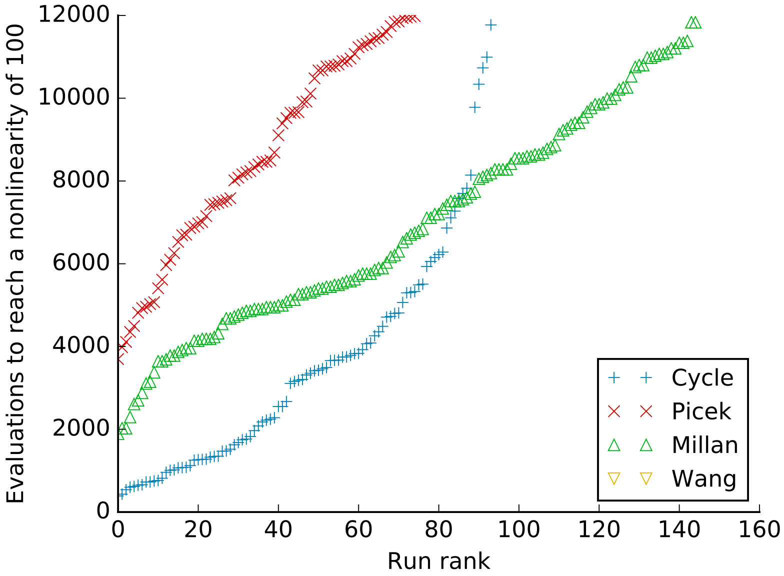

Figure 3: Number of evaluations until a run produces a nonlinearity of 100. We sort runs by rank, comparing the most efficient runs, the second most efficient runs, and so forth. Out of 1,000 runs, we only show those that reach a nonlinearity of 100. Cycle consistently outperforms Picek. While less efficient than Cycle, more Millan runs achieve a nonlinearity of 100. Lower is more efficient.

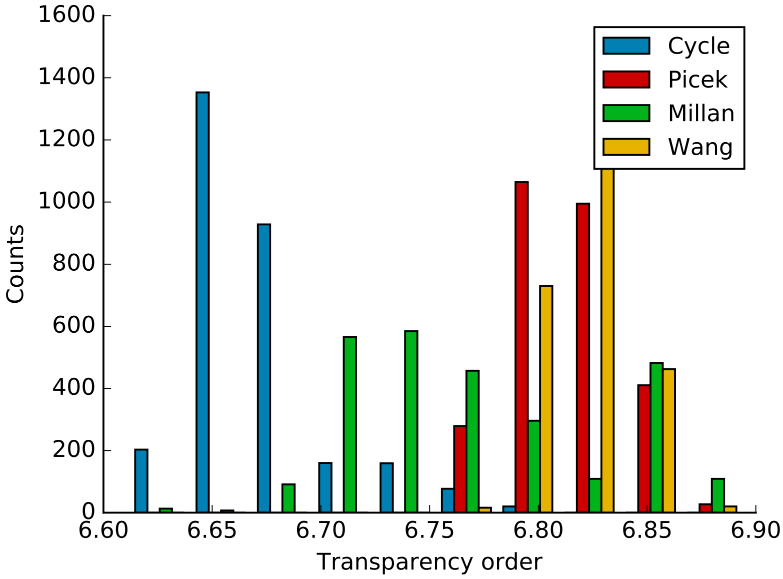

Figure 4: Transparency order histograms of four genetic algorithms. We include the 20 best observed individuals per run for 1,000 runs, excluding duplicates and S-boxes with fixed points. Farther left is better.

We collect the *hall of fame*, or 20 best individuals observed over a full evolution, from each of 1,000 independent runs. As the algorithms have as similar as possible number of evaluations[2] we find comparing the halls of fame fair. This produces 20,000 individuals per algorithm. We remove S-boxes with fixed points as unusable and remove duplicates to prevent double counting. We compare the halls of fame instead of the full final population. Genetic algorithms often explore bad search regions, which enables escape from local optima. We evaluate on the halls of fame to avoid penalizing this behavior.

Figure 4 shows the transparency order distributions of the halls of fame for the four algorithms tested. Cycle produces individuals with lower transparency orders than the other approaches. Millan produces the second best transparency orders, but the values exhibit a large spread. The best individuals produced by Picek and Wang are no better than the worst members of Cycle's halls of fame in transparency order.

Figure 5 shows the nonlinearity distributions of the halls of fame for the four algorithms. Cycle produces more hall of fame individuals with nonlinearity 100 than Millan, in contrast to runs (Figure 3). Millan produces more nonlinearity 100 individuals than Picek. As Wang failed to produce nonlinearity 100 runs, it lacks such individuals in its halls of fame. Wang performed strikingly worse than the other algorithms, often having nonlinearity 96, instead of 98.

To determine whether the differences observed in Figures 4 and 5 are statistically significant, we apply the Mann-Whitney U test [MW47]. This test enable us to determine whether an algorithm produces better values than another when the distributions are non-normal and discrete. To minimize the number of statistical tests we do, we compare closer distributions with each other. Cycle has a lower transparency order than Millan ($p \ll 0.001$), which has a lower transparency order than Picek ($p \ll 0.001$), which has a lower transparency order than Wang ($p \ll 0.001$). Cycle has a higher nonlinearities than Millan and Picek ($p = 0.01$, $p \ll 0.001$), which have higher nonlinearities than Wang

[2] Millan et al. has 12,015 instead of 12,000.

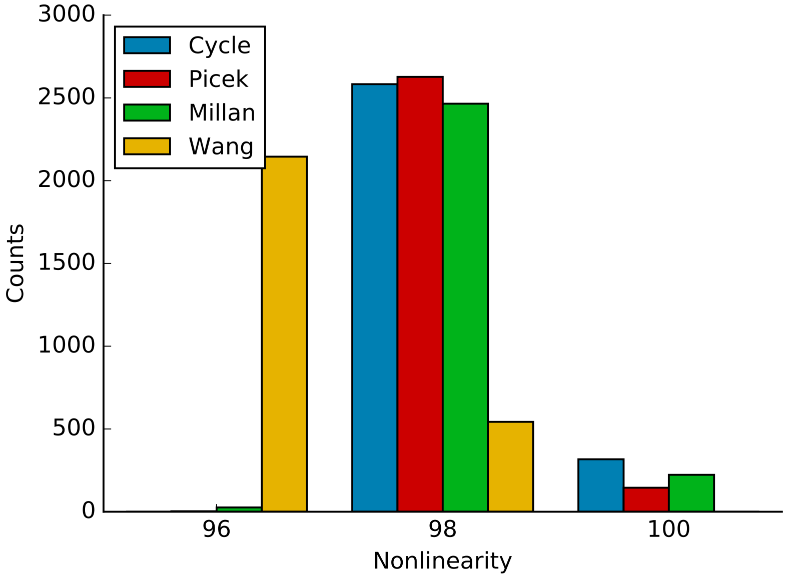

Figure 5: Nonlinearity histograms of four genetic algorithms. We include the 20 best observed individuals per run for 1,000 runs, excluding duplicates and S-boxes with fixed points. More Cycle individuals reach a nonlinearity of 100. Farther right is better.

($p \ll 0.001$, $p \ll 0.001$). Millan has higher nonlinearities than Picek, but the difference lacks statistical significance ($p = 0.08$).

4 Mitigating Fixed Points

A fixed point through a round function can break the security of a multi-round cipher. This section considers how to proactively avoid fixed points in S-boxes.

When generating S-boxes without regard for fixed points, we found that few S-boxes were free from (anti-)fixed points. Millan et al. yielded 14% of S-boxes without (anti-)fixed points, Wang et al. yielded 13%, Picek et al. yielded 14%, and Cycle yielded 15%. These results are no better than chance, since a uniformly-chosen 8-bit permutation lacks fixed points with probability 0.134 [SV16]. This section considers strategies for improving the yield, while attempting to maintain the other properties.

We compare four approaches to avoiding fixed points and anti-fixed points in S-boxes. The first approach, *Objective*, places a penalty in the fitness function. The second approach, *Iterative*, repairs fixed points directly (similar to bijectivity repair [WWLL12]). Specifically, we repair (anti-)fixed points by permuting values when there are multiple points. When there is a single fixed point or two fixed points that are the two's complements we add a random value into the exchange. We repeat until there are no fixed points. The third approach, *Final random*, repairs fixed points in the same way but only at the end of the evolution, instead of in every generation. The fourth approach, *Final single*, repairs single fixed points at the end of the evolution, choosing the best swap of the fixed point value. We built upon the Cycle approach, which we found most performant in Section 3.

We compare the halls of fame for Cycle, Objective, and Iterative. This produces 20,000 individuals per algorithm, except that we remove duplicates to prevent double counting and remove S-boxes with fixed points. For Final random and Final single we repair S-boxes from the Cycle halls of fame.

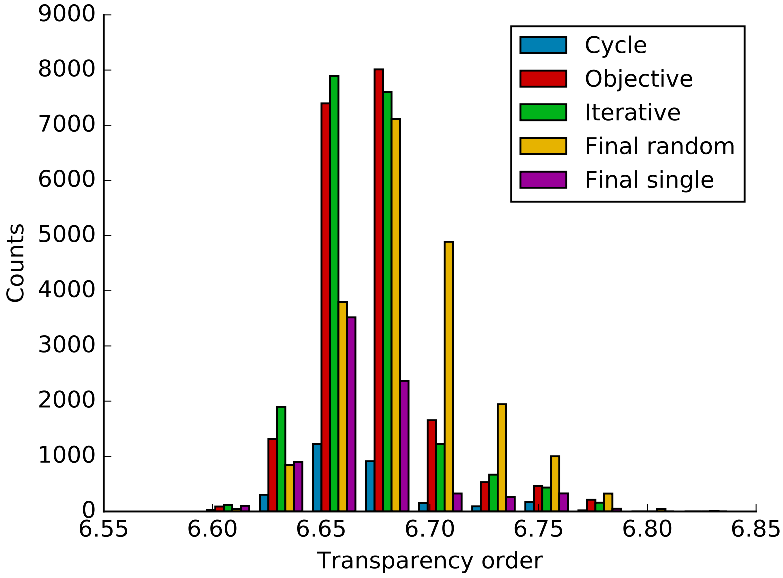

Figure 6: Transparency order histograms for Cycle and four approaches to avoiding fixed points. We restrict to S-boxes without fixed points. Farther left is better.

For these approaches we measure the number of fixed points, the transparency order, and the nonlinearity. We find 15% of the Cycle hall of fame S-boxes free of fixed points. All four repair approaches unsurprisingly reduce the number of fixed points; 98% of Objective, 100% of Iterative, 100% of Final random, 39% of Final single S-boxes were free of fixed points. Iterative and Final random lack fix points because they remove all fixed points. Final single removes the single fixed point from and S-box and ignores S-boxes with more fixed points. Specifically, we search for the best value to swap; this could be considered the Millan et al. hybrid approach applied to fixed points. Objective selects for S-boxes without fixed points, but they remain possible given the other components of the objective function.

Figure 6 shows the histograms for transparency order. The bin count records the yield out of 20,000 individuals at a particular transparency order value. We test whether the distributions differ for two approaches with the Mann-Whitney U test [MW47]. We find that Final single has lower transparency orders than Cycle ($p \ll 0.001$), which has lower transparency orders than Iterative ($p = 0.005$), which has lower transparency orders than Objective ($p \ll 0.001$), which has lower transparency orders than Final random ($p \ll 0.001$).

Figure 7 displays the nonlinearity histograms for each approach. We find no statistical difference between the nonlinearities of Cycle and Final single ($p = 0.27$) or the nonlinearities of Objective and Iterative ($p = 0.16$). We find Cycle and Final single to have higher nonlinearities than Objective ($p = 0.01$, $p = 0.01$) and Iterative ($p = 0.003$, $p \ll 0.001$), which have higher nonlinearities than Final random ($p \ll 0.001$, $p \ll 0.001$). These values are statistically significant under the Holm-Bonferroni correction.

These tests compare the distributions, but ignore the yield. Inspection of the figures reveals that if a higher yield is needed than Cycle provides, then Final single should be used. If a much larger yield is needed, then Iterative should be used.

Like Cycle, Objective makes 12,000 fitness evaluations per run, but on an objective function that includes a penalty for fixed points. Iterative performs the same number of evaluations, but S-boxes with fixed points undergo random swaps until free of fixed points.

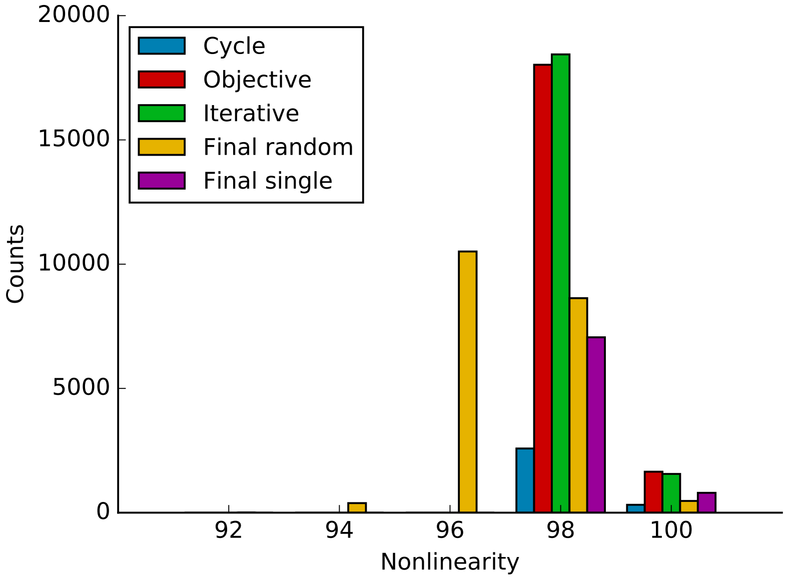

Figure 7: Nonlinearity histograms for Cycle and four approaches to avoiding fixed points. We restrict to S-boxes without fixed points. Farther right is better.

The approaches that repair at the end build upon a preexisting hall of fame. The random repair approach makes no additional evaluations. The single repair approach makes 254 evaluations per individual with a single fixed point (which accounted for 25% of the hall of fame).

5 Conclusion

This research aimed to take a rigorous approach to evolving 8-bit S-boxes with good cryptanalytic and differential power analysis (DPA) properties.

We estimated the trade-off between nonlinearity and transparency order and represented it as a Pareto front. To accomplish this, we were the first to apply multiobjective evolutionary algorithms to S-box generation. We found SPEA2 to perform slightly better than NSGA-II. The multiobjective algorithms produced solutions superior to manually designed S-boxes (AES, SAFER, Camellia, and E2). Random permutations attained a narrow range of transparency orders. We observed a tight relationship between nonlinearity and transparency order; the nonlinearity drops substantially before the transparency order does. Unfortunately, this suggests the difficulty of finding S-boxes with high nonlinearity and low transparency order, if they exist.

We showed that the cycle crossover together with the swap mutation yields more efficient genetic algorithms for generating S-boxes, both in number of good solutions and speed to attain them. We demonstrated the efficiency of the cycle crossover in the first side-by-side comparison of the genetic algorithms of Millan et al. [MBC+99], Wang et al. [WWLL12], and Picek et al. [PMMB15]. The cycle crossover outperformed Millan, which outperformed Picek, which outperformed Wang.

We proposed and compared several methods for finding S-boxes without fixed points. We found that swapping a single fixed point with the best value on the final S-box produces the best S-boxes, while random swaps to remove fixed points at the end produce the worst S-boxes. Repairing fixed points during evolution or including a fixed point penalty in the objective function produced a higher yield of good S-boxes, but a worse overall distribution.

We propose two directions for future work. First, our multiobjective evolutionary algorithm should be extended to capture several cryptanalytic metrics simultaneously; concretely, we plan to add support for delta uniformity and autocorrelation resistance. Second, extending transparency order (which represents noiseless measurements) with an analytical framework for noise would enable an argument about the relative impact of collecting an additional trace versus capturing an extra plaintext/ciphertext pair, so that the defender may have concrete guidance about the appropriate S-box to choose along the Pareto front.

Acknowledgments.

We gratefully acknowledge the support of Sukarno Mertoguno in the Office of Naval Research. The second author also acknowledges NSF grant 1414119. Additionally, we thank Rob Cunningham.

References

[AIK$^+$00] Kazumaro Aoki, Tetsuya Ichikawa, Masayuki Kanda, Mitsuru Matsui, Shiho Moriai, Junko Nakajima, and Toshio Tokita. Camellia: A 128-bit block cipher suitable for multiple platforms - design and analysis. In *SAC*, volume 2012 of *LNCS*, pages 39–56. Springer, 2000.

[BAK98] Eli Biham, Ross J. Anderson, and Lars R. Knudsen. Serpent: A new block cipher proposal. In *FSE*, volume 1372 of *LNCS*, pages 222–238. Springer, 1998.

[BCD$^+$98] Carolynn Burwick, Don Coppersmith, Edward D'Avignon, Rosario Gennaro, Shai Halevi, Charanjit Jutla, Stephen M Matyas Jr, Luke O'Connor, Mohammad Peyravian, David Safford, et al. Mars-a candidate cipher for aes. *NIST AES Proposal*, 268, 1998.

[BCG$^+$12] Julia Borghoff, Anne Canteaut, Tim Güneysu, Elif Bilge Kavun, Miroslav Knezevic, Lars R. Knudsen, Gregor Leander, Ventzislav Nikov, Christof Paar, Christian Rechberger, Peter Rombouts, Søren S. Thomsen, and Tolga Yalçin. PRINCE - A low-latency block cipher for pervasive computing applications - extended abstract. In *ASIACRYPT*, volume 7658 of *LNCS*, pages 208–225. Springer, 2012.

[BG09] Céline Blondeau and Benoît Gérard. On the data complexity of statistical attacks against block ciphers. In *Workshop on Coding and Cryptography-WCC*, volume 2009, pages 469–488, 2009.

[BGT11] Céline Blondeau, Benoît Gérard, and Jean-Pierre Tillich. Accurate estimates of the data complexity and success probability for various cryptanalyses. *Designs, Codes and Cryptography*, 59(1-3):3–34, 2011.

[BKL$^+$07] Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin, and C. Vikkelsoe. PRESENT: an ultra-lightweight block cipher. In *CHES*, volume 4727 of *LNCS*, pages 450–466. Springer, 2007.

[Bor00] J Borst. The block cipher: Grand cru. *Primitive submitted to NESSIE*, page 4, 2000.

[BR00] Paulo Barreto and Vincent Rijmen. The anubis block cipher. *Submission to the NESSIE Project*, 2000.

[BRSS10] John Black, Phillip Rogaway, Thomas Shrimpton, and Martijn Stam. An analysis of the blockcipher-based hash functions from PGV. *J. Cryptology*, 23(4):519–545, 2010.

[CF04] Hua Chen and Deng-Guo Feng. An effective evolutionary strategy for bijective S-boxes. In *Evolutionary Computation, 2004. CEC2004. Congress on*, volume 2, pages 2120–2123. IEEE, 2004.

[CMS$^+$14] Kaushik Chakraborty, Subhamoy Maitra, Sumanta Sarkar, Bodhisatwa Mazumdar, Debdeep Mukhopadhyay, and Emmanuel Prouff. Redefining the transparency order. Technical report, Cryptology ePrint Archive, Report 2014/367, 2014.

[CSM$^+$15] K. Chakraborty, S. Sarkar, S. Maitra, B. Mazumdar, D. Mukhopadhyay, and E Prouff. Redefining the transparency order. In *Coding and Cryptography, International Workshop*, 2015.

[DAPM00] Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and Tanaka Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In *Parallel problem solving from nature PPSN VI*, pages 849–858. Springer, 2000.

[DC07] Christophe De Canniere. Analysis and design of symmetric encryption algorithms. *Doctoral Dissertaion, KULeuven*, 2007.

[DEMS14] Christoph Dobraunig, Maria Eichlseder, Florian Mendel, and Martin Schläffer. Ascon v1. *CAESAR Competition*, 2014.

[DJ75] Kenneth Alan De Jong. *An Analysis of the Behavior of a Class of Genetic Adaptive Systems.* PhD thesis, Ann Arbor, MI, USA, 1975. AAI7609381.

[DR02] Joan Daemen and Vincent Rijmen. *The Design of Rijndael.* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2002.

[Dwoa] Morris Dworkin. Recommendation for block cipher modes of operation: The CCM mode for authentication and confidentiality. NIST SP 800-38C.

[Dwob] Morris Dworkin. Recommendation for block cipher modes of operation: The CMAC mode for authentication. NIST SP 800-38B.

[Dwo01] Morris Dworkin. Recommendation for block cipher modes of operation: Methods and techniques. NIST SP 800-38A, 2001.

[Dwo07] Morris Dworkin. Recommendation for block cipher modes of operation: Galois/counter mode (GCM) and GMAC. NIST SP 800-38D, November 2007.

[Dwo10] Morris Dworkin. Recommendation for block cipher modes of operation: The XTS-AES mode for confidentiality on storage devices. NIST SP 800-38E, January 2010.

[FDG$^+$12] Félix-Antoine Fortin, François-Michel De Rainville, Marc-André Gardner, Marc Parizeau, and Christian Gagné. DEAP: Evolutionary algorithms made easy. *Journal of Machine Learning Research*, 13:2171–2175, jul 2012.

[FLD12] Yunsi Fei, Qiasi Luo, and A Adam Ding. A statistical model for DPA with novel algorithmic confusion analysis. In *CHES*, pages 233–250. Springer, 2012.

[Gol89] David E Goldberg. *Genetic algorithms in search and machine learning.* Addison Wesley, 1989.

[HKR15] Viet Tung Hoang, Ted Krovetz, and Phillip Rogaway. Robust authenticated-encryption AEZ and the problem that it solves. In *EUROCRYPT*, volume 9056 of *LNCS*, pages 15–44. Springer, 2015.

[Hol68] J. H. Holland. Hierarchical descriptions of universal spaces and adaptive systems. Technical Report ORA Projects 01252 and 08226, University of Michigan, Dept. of Computer Science and Communication Sciences, 1968.

[INN14] Georgi Ivanov, Nikolay Nikolov, and Svetla Nikova. Reversed genetic algorithms for generation of bijective s-boxes with good cryptographic properties. *Cryptography and Communications*, pages 1–30, 2014.

[Jaf07] Joshua Jaffe. A first-order DPA attack against AES in counter mode with unknown initial counter. In *CHES*, volume 4727 of *LNCS*, pages 1–13. Springer, 2007.

[JNP14] Jérémy Jean, Ivica Nikolic, and Thomas Peyrin. Tweaks and keys for block ciphers: The TWEAKEY framework. In *ASIACRYPT*, volume 8874 of *LNCS*, pages 274–288. Springer, 2014.

[KJJ99] Paul C. Kocher, Joshua Jaffe, and Benjamin Jun. Differential power analysis. In *CRYPTO*, pages 388–397, London, UK, 1999. Springer-Verlag.

[KMTM00] Masayuki Kanda, Shiho Moriai, Youichi Takashima, and Tsutomu Matsumoto. E2–a new 128-bit block cipher. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 83(1):48–59, 2000.

[KR11] Lars R. Knudsen and Matthew Robshaw. *The Block Cipher Companion.* Information Security and Cryptography. Springer, 2011.

[Lim99] Chae Hoon Lim. A revised version of crypton - crypton V1.0. In *FSE*, volume 1636 of *LNCS*, pages 31–45. Springer, 1999.

[LPPS07] Gregor Leander, Christof Paar, Axel Poschmann, and Kai Schramm. New lightweight DES variants. In *FSE*, volume 4593 of *LNCS*, pages 196–210. Springer, 2007.

[Mas93] James L. Massey. SAFER K-64: A byte-oriented block-ciphering algorithm. In *FSE*, volume 809 of *LNCS*, pages 1–17. Springer, 1993.

[MBC$^+$99] William Millan, L. Burnett, Gary Carter, Andrew J. Clark, and Ed Dawson. Evolutionary heuristics for finding cryptographically strong S-boxes. In *Information and Communication Security*, volume 1726 of *LNCS*, pages 263–274. Springer, 1999.

[Mer90] Ralph C. Merkle. Fast software encryption functions. In *CRYPTO*, volume 537 of *LNCS*, pages 476–501. Springer, 1990.

[MMS13] Bodhisatwa Mazumdar, Debdeep Mukhopadhyay, and Indranil Sengupta. Constrained search for a class of good bijective-boxes with improved DPA resistivity. *Information Forensics and Security, IEEE Transactions on*, 8(12):2154–2163, 2013.

[MW47] Henry B Mann and Donald R Whitney. On a test of whether one of two random variables is stochastically larger than the other. *The annals of mathematical statistics*, pages 50–60, 1947.

[Nat77] National Institute of Standards and Technology. Data Encryption Standard (DES). In *Federal Information Processing Standards Publication*, volume 46. National Bureau of Standards, January 1977.

[Nat01] National Institute of Standards and Technology. Federal Information Processing Standards Publication 197: Announcing the Advanced Encryption Standard, November 2001.

[Nik14] Ivica Nikolic. Tiaoxin – 346. *Submission to the CAESAR Competition*, 2014.

[OSH87] I Oliver, D J Smith, and John R C Holland. Study of permutation crossover operators on the traveling salesman problem. In *Genetic algorithms and their applications: proceedings of the second International Conference on Genetic Algorithms: July 28-31, 1987 at the Massachusetts Institute of Technology, Cambridge, MA*. Hillsdale, NJ: L. Erlhaum Associates, 1987., 1987.

[PEB$^+$14] Stjepan Picek, Bariş Ege, Lejla Batina, Domagoj Jakobovic, Łukasz Chmielewski, and Marin Golub. On using genetic algorithms for intrinsic side-channel resistance: the case of AES S-box. In *Cryptography and Security in Computing Systems*, pages 13–18. ACM, 2014.

[PEP$^+$14] Stjepan Picek, Baris Ege, Kostas Papagiannopoulos, Lejla Batina, and Domagoj Jakobovic. Optimality and beyond: The case of 4× 4 S-boxes. In *Hardware-Oriented Security and Trust, IEEE International Symposium on*, 2014.

[PGV93] Bart Preneel, René Govaerts, and Joos Vandewalle. Hash functions based on block ciphers: A synthetic approach. In *CRYPTO*, volume 773 of *LNCS*, pages 368–378. Springer, 1993.

[PMMB15] Stjepan Picek, Bodhisatwa Mazumdar, Debdeep Mukhopadhyay, and Lejla Batina. Modified transparency order property: solution or just another attempt. In *Security, Privacy, and Applied Cryptography Engineering*, pages 210–227. Springer, 2015.

[PPE$^+$14] Stjepan Picek, Kostas Papagiannopoulos, Barış Ege, Lejla Batina, and Domagoj Jakobovic. Confused by confusion: Systematic evaluation of DPA resistance of various S-boxes. In *INDOCRYPT*, pages 374–390. Springer, 2014.

[Sch93] Bruce Schneier. Description of a new variable-length key, 64-bit block cipher (blowfish). In *FSE*, volume 809 of *LNCS*, pages 191–204. Springer, 1993.

[SKW$^+$99] Bruce Schneier, John Kelsey, Doug Whiting, David Wagner, Chris Hall, and Niels Ferguson. *The Twofish encryption algorithm: a 128-bit block cipher*. John Wiley & Sons, Inc., 1999.

[SMMK12] Tomoyasu Suzaki, Kazuhiko Minematsu, Sumio Morioka, and Eita Kobayashi. TWINE: A lightweight block cipher for multiple platforms. In *Selected Areas in Cryptography, 19th International Conference, SAC 2012, Windsor, ON, Canada, August 15-16, 2012, Revised Selected Papers*, volume 7707 of *LNCS*, pages 339–354. Springer, 2012.

[SV16] Merrielle Spain and Mayank Varia. Diversity within the Rijndael design principles for resistance to differential power analysis. In *International Conference on Cryptology and Network Security*. Springer, 2016.

[Wil45] Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics bulletin*, 1(6):80–83, 1945.

[WWLL12] Yong Wang, Kwok-Wo Wong, Changbing Li, and Yang Li. A novel method to design s-box based on chaotic map and genetic algorithm. *Physics Letters A*, 376(6):827–833, 2012.

[ZLT01] Eckart Zitzler, Marco Laumanns, and Lothar Thiele. SPEA2: Improving the strength pareto evolutionary algorithm. Technical report, 2001.